Task_3_Prodigy_internship

June 4, 2024

PRODIGY INFOTECH DATA SCIENCE INTERN

#TASK 3

TASK OVERVIEW: Build a decision tree classifier to predict whether a customer will purchase a product or service based on their demographic and behavioral data. Use a dataset such as the Bank Marketing dataset from the UCI Machine Learning Repository.

IMPORTING THE LIBRARIES AND DATASET

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
```

[ ]:
```
df = pd.read_csv("/content/bank marketing dataset.csv")
```

DATA PREPROCCESSING

[ ]:
```
df.head()
```

[ ]:
```
   age         job  marital  education default  balance housing loan  contact  \
0   59      admin.  married  secondary      no     2343     yes   no  unknown
1   56      admin.  married  secondary      no       45      no   no  unknown
2   41  technician  married  secondary      no     1270     yes   no  unknown
3   55    services  married  secondary      no     2476     yes   no  unknown
4   54      admin.  married   tertiary      no      184      no   no  unknown

   day month  duration  campaign  pdays  previous poutcome deposit
0    5   may      1042         1     -1         0  unknown     yes
1    5   may      1467         1     -1         0  unknown     yes
2    5   may      1389         1     -1         0  unknown     yes
3    5   may       579         1     -1         0  unknown     yes
4    5   may       673         2     -1         0  unknown     yes
```

[ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11162 entries, 0 to 11161
```

```
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        11162 non-null  int64 
 1   job        11162 non-null  object
 2   marital    11162 non-null  object
 3   education  11162 non-null  object
 4   default    11162 non-null  object
 5   balance    11162 non-null  int64 
 6   housing    11162 non-null  object
 7   loan       11162 non-null  object
 8   contact    11162 non-null  object
 9   day        11162 non-null  int64 
 10  month      11162 non-null  object
 11  duration   11162 non-null  int64 
 12  campaign   11162 non-null  int64 
 13  pdays      11162 non-null  int64 
 14  previous   11162 non-null  int64 
 15  poutcome   11162 non-null  object
 16  deposit    11162 non-null  object
dtypes: int64(7), object(10)
memory usage: 1.4+ MB
```

[ ]:
```
df.nunique()
```

[ ]:
```
age            76
job            12
marital         3
education       4
default         2
balance      3805
housing         2
loan            2
contact         3
day            31
month          12
duration     1428
campaign       36
pdays         472
previous       34
poutcome        4
deposit         2
dtype: int64
```

[ ]:
```
df.duplicated()
```

[ ]:
```
0        False
1        False
2        False
3        False
4        False
         …
11157    False
11158    False
11159    False
11160    False
11161    False
Length: 11162, dtype: bool
```

[ ]:
```
df.columns
```

[ ]:
```
Index(['age', 'job', 'marital', 'education', 'default', 'balance', 'housing',
       'loan', 'contact', 'day', 'month', 'duration', 'campaign', 'pdays',
       'previous', 'poutcome', 'deposit'],
      dtype='object')
```

[ ]:
```
df.isnull().sum()
```

[ ]:
```
age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
deposit      0
dtype: int64
```

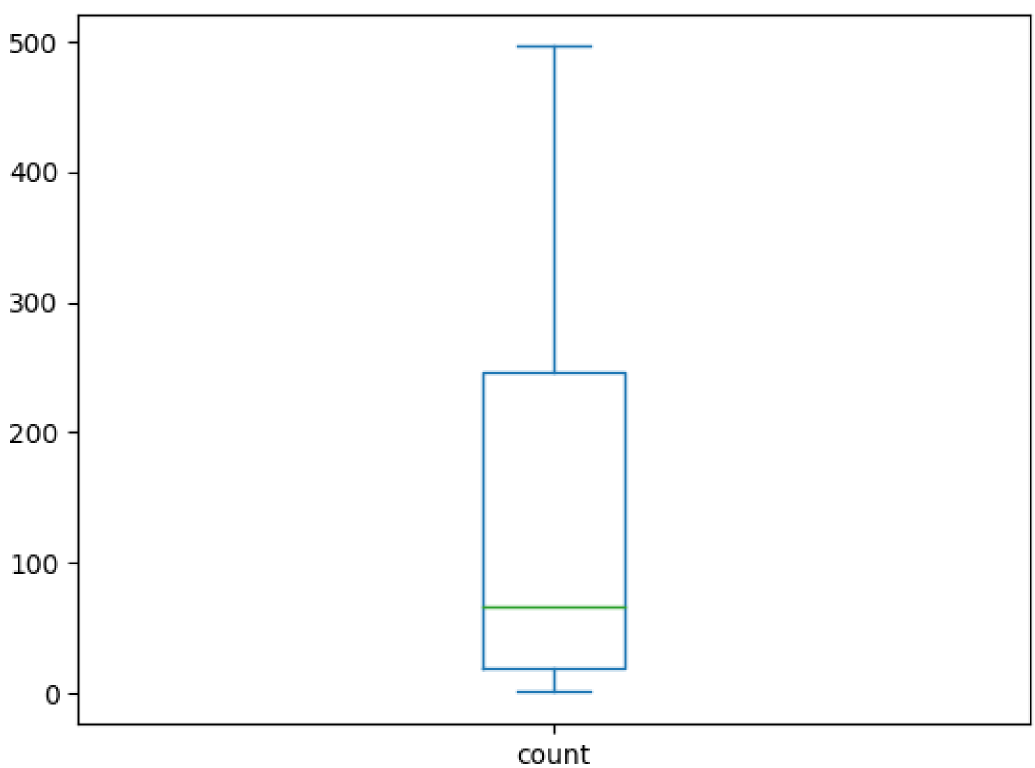
EDA

[ ]:
```
age_value = df['age'].value_counts()
age_value
```

[ ]:
```
age
31    496
32    477
34    466
33    464
35    461
     …
92      2
93      2
88      2
95      1
89      1
Name: count, Length: 76, dtype: int64
```

[ ]:
```
age_value.plot(kind="box")
```

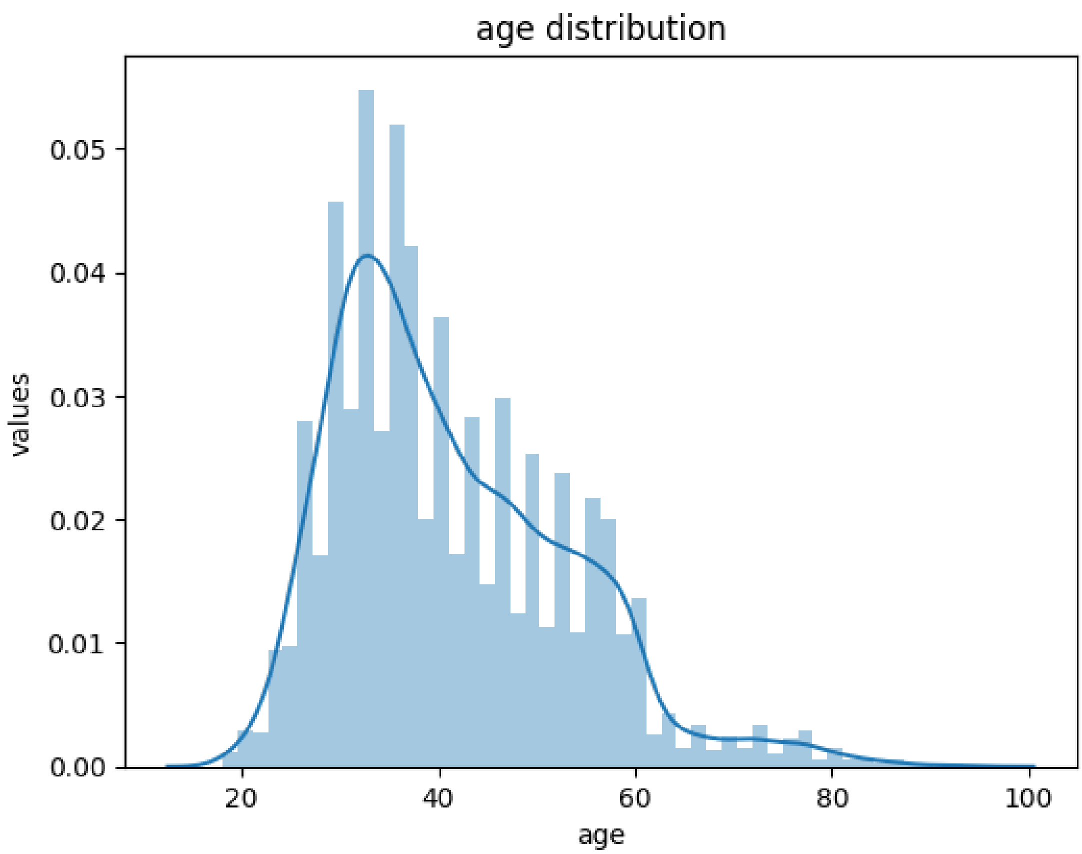
[ ]: <Axes: >



[ ]:
```
sns.distplot(df['age'])
plt.title("age distribution")
plt.xlabel("age")
plt.ylabel('values')
```

```
&lt;ipython-input-12-f0501e0852cd&gt;:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['age'])
```

[ ]: Text(0, 0.5, 'values')



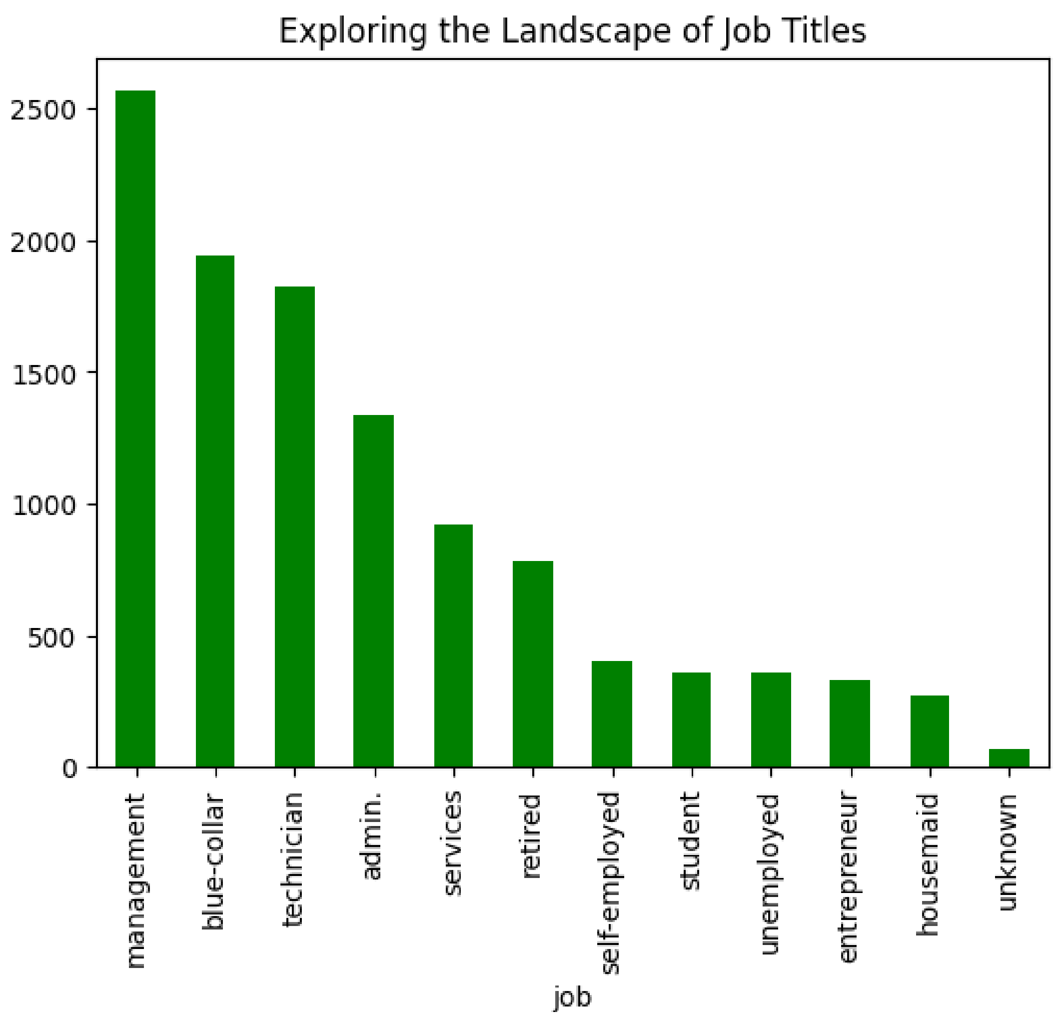
[ ]:
```
job = df['job'].value_counts()
job
```

[ ]:
```
job
management       2566
blue-collar      1944
```

```
technician       1823
admin.           1334
services          923
retired           778
self-employed     405
student           360
unemployed        357
entrepreneur      328
housemaid         274
unknown            70
Name: count, dtype: int64
```

[ ]:
```
job.plot(kind="bar",color='green')
plt.title("Exploring the Landscape of Job Titles")
```

[ ]: `Text(0.5, 1.0, 'Exploring the Landscape of Job Titles')`

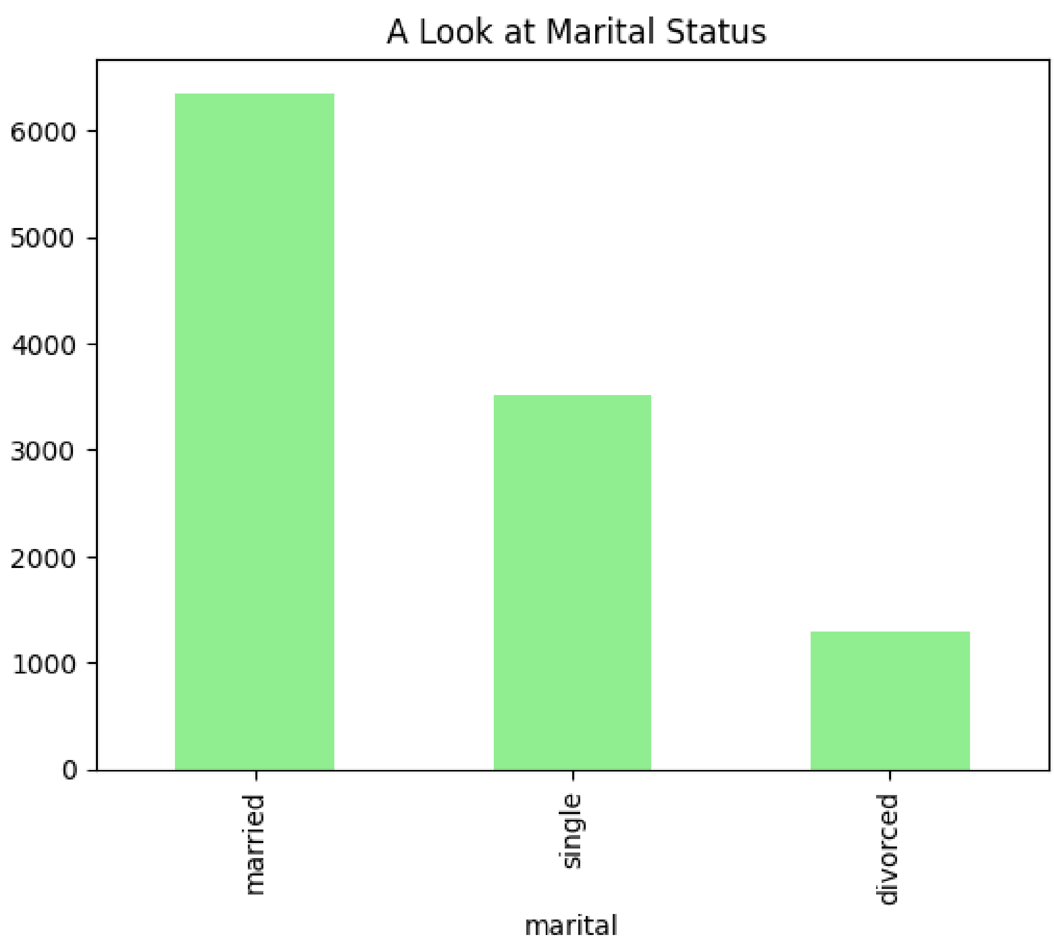


[ ]:
```
marital_status = df['marital'].value_counts()
marital_status
```

[ ]:
```
marital
married     6351
single      3518
divorced    1293
Name: count, dtype: int64
```

[ ]:
```
marital_status.plot(kind="bar",color='lightgreen')
plt.title(" A Look at Marital Status")
```

[ ]:
```
Text(0.5, 1.0, ' A Look at Marital Status')
```

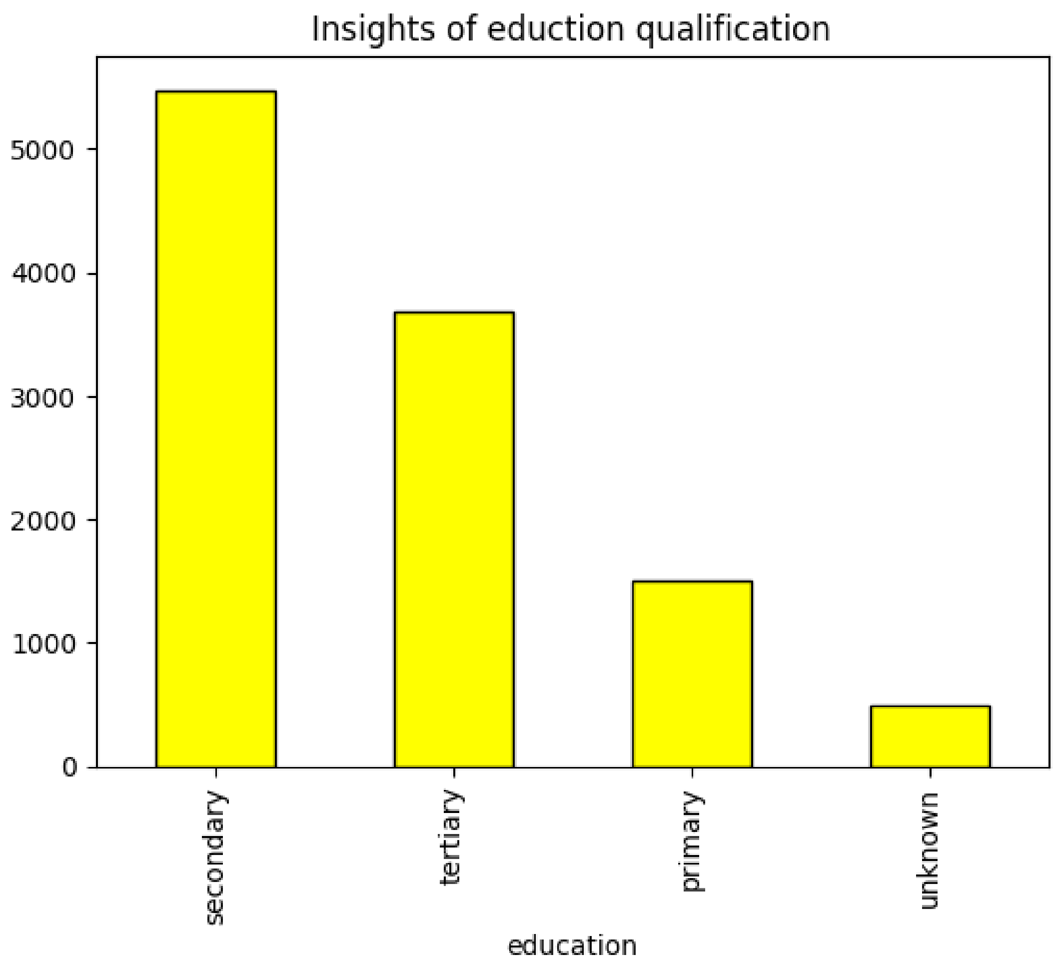


[ ]:
```
education_distribution = df['education'].value_counts()
education_distribution
```

[ ]: education
    secondary   5476
    tertiary   3689
    primary   1500
    unknown   497
    Name: count, dtype: int64

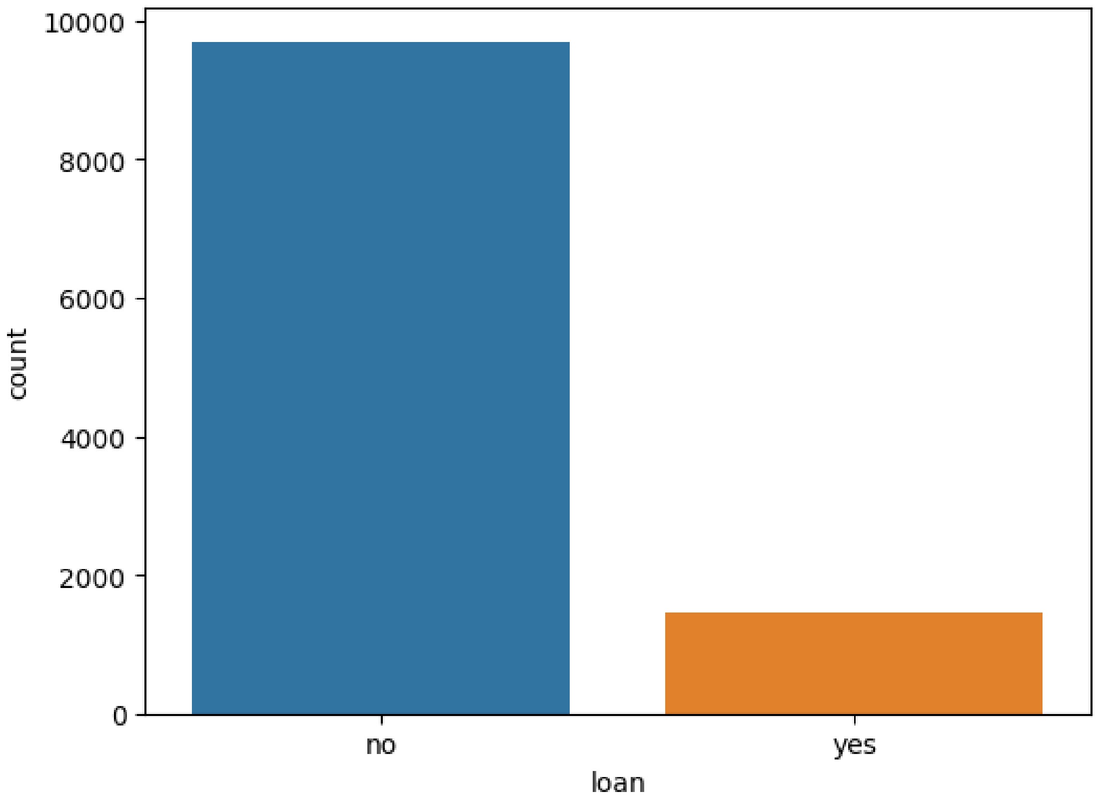
```
[ ]: education_distribution.plot(kind="bar",color="yellow",edgecolor="black")
     plt.title("Insights of eduction qualification")
```

[ ]: Text(0.5, 1.0, 'Insights of eduction qualification')



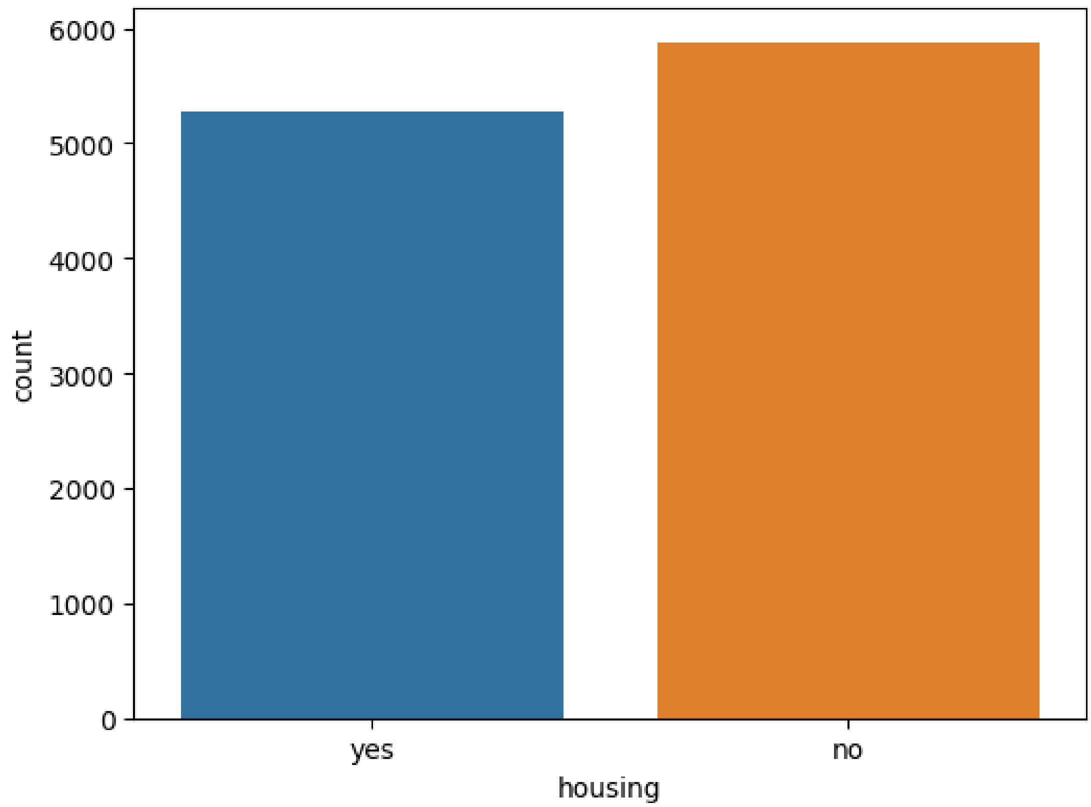

```
[ ]: sns.countplot(x="loan",hue="loan",data=df)
```

[ ]: <Axes: xlabel='loan', ylabel='count'>



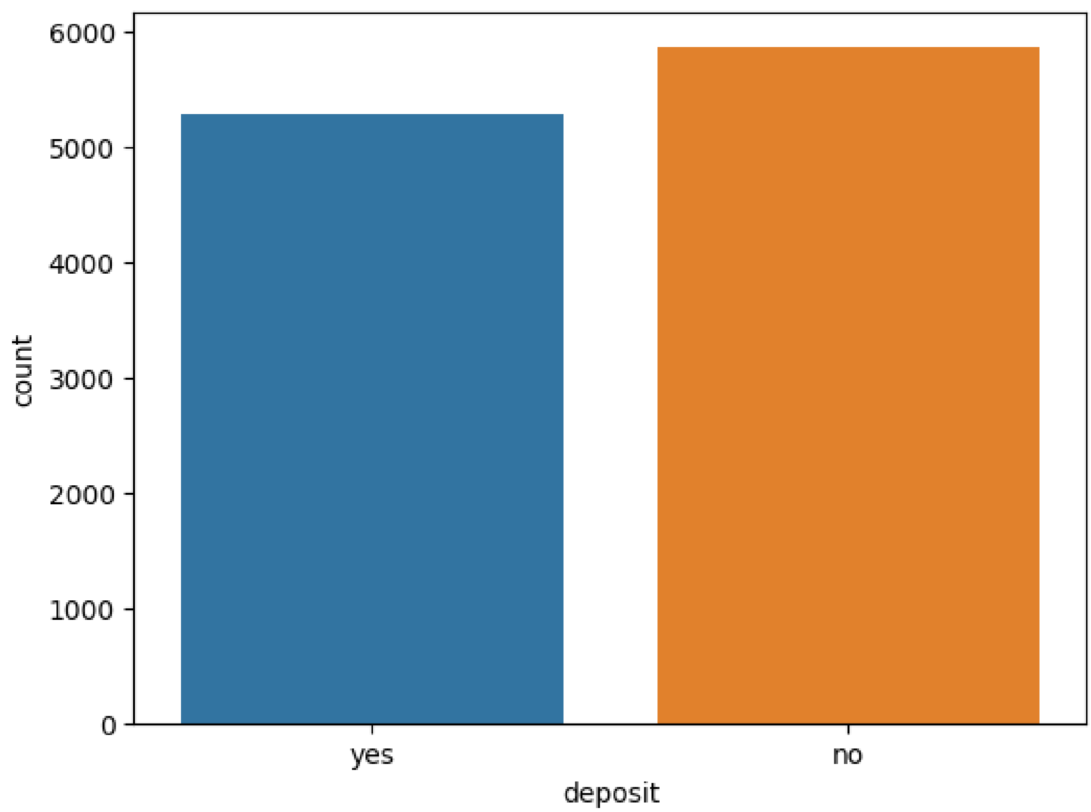

```
[ ]: sns.countplot(x="housing",hue="housing",data=df)
```

```
[ ]: <Axes: xlabel='housing', ylabel='count'>
```



[ ]:
```
sns.countplot(x="deposit",hue="deposit",data=df)
```

[ ]: <Axes: xlabel='deposit', ylabel='count'>



Feature Engineering

[ ]:
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder


# Using Label Encoder
label_encoder = LabelEncoder()
df_label_encoded = df.copy()
df_label_encoded["job"] = label_encoder.fit_transform(df["job"])
df_label_encoded["marital"] = label_encoder.fit_transform(df["marital"])
df_label_encoded["education"] = label_encoder.fit_transform(df["education"])
df_label_encoded["default"] = label_encoder.fit_transform(df["default"])
df_label_encoded["deposit"] = label_encoder.fit_transform(df["deposit"])
df_label_encoded["loan"] = label_encoder.fit_transform(df["loan"])
df_label_encoded["housing"] = label_encoder.fit_transform(df["housing"])
print("Label Encoded Data:")
print(df_label_encoded)
```

```
Label Encoded Data:
       age  job  marital  education  default  balance  housing  loan  \
0       59    0        1          1        0     2343        1     0
```

```
1        56    0         1           1         0       45         0      0
2        41    9         1           1         0     1270         1      0
3        55    7         1           1         0     2476         1      0
4        54    0         1           1         0      184         0      0
…    …   …      …         …      …      …      …     …
11157    33    1         2           0         0        1         1      0
11158    39    7         1           1         0      733         0      0
11159    32    9         2           1         0       29         0      0
11160    43    9         1           1         0        0         0      1
11161    34    9         1           1         0        0         0      0

        contact  day month  duration  campaign  pdays  previous poutcome  \
0       unknown    5   may      1042         1     -1         0  unknown
1       unknown    5   may      1467         1     -1         0  unknown
2       unknown    5   may      1389         1     -1         0  unknown
3       unknown    5   may       579         1     -1         0  unknown
4       unknown    5   may       673         2     -1         0  unknown
…           …   …   …        …       …     …       …      …
11157  cellular   20   apr       257         1     -1         0  unknown
11158   unknown   16   jun        83         4     -1         0  unknown
11159  cellular   19   aug       156         2     -1         0  unknown
11160  cellular    8   may         9         2    172         5  failure
11161  cellular    9   jul       628         1     -1         0  unknown

       deposit
0            1
1            1
2            1
3            1
4            1
…        …
11157        0
11158        0
11159        0
11160        0
11161        0

[11162 rows x 17 columns]
```

[ ]:
```
df_label_encoded.
 ↪drop(columns=['contact','day','month','duration','poutcome'],inplace=True)
```

[ ]:
```
df_label_encoded
```

[ ]:
```
       age  job  marital  education  default  balance  housing  loan  \
0       59    0        1          1        0     2343        1     0
1       56    0        1          1        0       45        0     0
```

```
2        41    9          1          1         0     1270        1     0
3        55    7          1          1         0     2476        1     0
4        54    0          1          2         0      184        0     0
…     …  …        …         …        …        …        …   …
11157    33    1          2          0         0        1        1     0
11158    39    7          1          1         0      733        0     0
11159    32    9          2          1         0       29        0     0
11160    43    9          1          1         0        0        0     1
11161    34    9          1          1         0        0        0     0

       campaign  pdays  previous  deposit
0             1     -1         0        1
1             1     -1         0        1
2             1     -1         0        1
3             1     -1         0        1
4             2     -1         0        1
…          …     …        …       …
11157         1     -1         0        0
11158         4     -1         0        0
11159         2     -1         0        0
11160         2    172         5        0
11161         1     -1         0        0

[11162 rows x 12 columns]
```

Train test Spliting

[ ]:
```
x = df_label_encoded.drop("deposit",axis=1)
```

[ ]:
```
x
```

[ ]:
```
       age  job  marital  education  default  balance  housing  loan  \
0        59    0          1          1         0     2343        1     0
1        56    0          1          1         0       45        0     0
2        41    9          1          1         0     1270        1     0
3        55    7          1          1         0     2476        1     0
4        54    0          1          2         0      184        0     0
…     …  …        …         …        …        …        …   …
11157    33    1          2          0         0        1        1     0
11158    39    7          1          1         0      733        0     0
11159    32    9          2          1         0       29        0     0
11160    43    9          1          1         0        0        0     1
11161    34    9          1          1         0        0        0     0

       campaign  pdays  previous
0             1     -1         0
1             1     -1         0
```

```
2               1     -1          0
3               1     -1          0
4               2     -1          0
…           …     …        …
11157           1     -1          0
11158           4     -1          0
11159           2     -1          0
11160           2    172          5
11161           1     -1          0

[11162 rows x 11 columns]
```

[ ]:
```
y = df_label_encoded['deposit']
```

[ ]:
```
y
```

[ ]:
```
0        1
1        1
2        1
3        1
4        1
        ..
11157    0
11158    0
11159    0
11160    0
11161    0
Name: deposit, Length: 11162, dtype: int64
```

[ ]:
```
x_train, x_test, y_train, y_test = train_test_split (x,y,test_size=0.
↪2,random_state=2)
```

[ ]:
```
x_train.shape
```

[ ]:
```
(8929, 11)
```

[ ]:
```
x_test.shape
```

[ ]:
```
(2233, 11)
```

[ ]:
```
y_train.shape
```

[ ]:
```
(8929,)
```

[ ]:
```
y_test.shape
```

[ ]:
```
(2233,)
```

MODEL TRAINING

[ ]:
```
from sklearn.tree import DecisionTreeClassifier
dt_model = DecisionTreeClassifier()
```

[ ]:
```
dt_model.fit(x_train,y_train)
```

[ ]:
```
DecisionTreeClassifier()
```

[ ]:
```
from sklearn.metrics import confusion_matrix, accuracy_score ,␣
↪classification_report
```

MODEL EVALUATION

[ ]:
```
confusion_matrix = pd.
↪DataFrame(confusion_matrix(y_test,y_pred),columns=['Predicted No','Predicted␣
↪Yes'],index=['Actual No','Actual Yes'])
```

[ ]:
```
confusion_matrix
```

[ ]:

	Predicted No	Predicted Yes
Actual No	711	476
Actual Yes	458	588

[ ]:
```
y_pred = dt_model.predict(x_test)
```

[ ]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.61      0.60      0.60      1187
           1       0.55      0.56      0.56      1046

    accuracy                           0.58      2233
   macro avg       0.58      0.58      0.58      2233
weighted avg       0.58      0.58      0.58      2233
```

[ ]:
```
print("Decision Tree Classifier")
print(f"Accuracy:{accuracy_score(y_test,y_pred)}",)
```

```
Decision Tree Classifier
Accuracy:0.5817286162113748
```